3D Human Limb Detection using Space Carving and Multi-view Eigen Models

Sidharth Bhatia* **Leonid Sigal*** **Michael Isard** † **Michael J. Black***

* Department of Computer Science, Brown University Providence, RI 02912
† Microsoft Research Silicon Valley, Mountain View, CA 94043
{sibhatia,ls,black}@cs.brown.edu, misard@microsoft.com

Abstract

In this paper, we integrate space carving and eigen detection methods to develop a bottom-up 3D human limb detector. We model the body in terms of its constituent body parts; here we focus on the head, lower arms, upper arms and calves. For each body part, we build a multi-view eigen model that combines image views from multiple calibrated cameras. This approach is much more constraining than the conventional multiple single-view eigen models and provides coarse 3D pose information. We use ideas from space carving using multiple silhouette images to constrain the volume of our search for the body part locations. We have applied the method to detect the body parts of a subject in long test sequences. The approach provides bottom-up information that supports the automatic initialization of a full 3D human body model.

1. Introduction

3D markerless motion capture and tracking is a complicated problem that has occupied the interest of the computer vision community for the past two decades. The problem is challenging due to high dimensionality of the articulated human body, the lack of explicit depth information in 2D images, self occlusion, complex human dynamics, and singularities in kinematics. A non-invasive and cheap motion tracking methodology would have applications in the fields of surveillance, human recognition through gait analysis, sports and rehabilitation medicine, motion pictures and gaming. One of the key challenges in 3D human tracking involves the automatic initialization of the body model or its re-initialization after tracking failure. Given the dimensionality of the body model we argue that direct top down search is impractical and instead focus on the bottom-up initialization of the model from low-level cues. Recent work [12] has shown that Bayesian inference in the form of non-parametric belief propagation [6] can be used to combine information from low-level part detectors to infer the configuration of the human body. Here we develop a method for detecting these body parts.

This paper develops a bottom-up body part detector that draws inspiration from work done by Cheung et al [2] and Ioffe and Forsyth [5]. In particular, we model the body in terms of its parts (here we consider the head, upper and lower limbs as in [5]). Unlike the work of Ioffe and Forsyth or Ramanan and Forsyth [8] which detect 2D parts, here we focus on determining the approximate 3D pose for each part. To simplify the search problem we exploit a simple learned prior model that makes use of space carving to eliminate large regions of the 3D volume as explored in [2]. Finally, we implement a multi-view eigen space detection method for determining the 3D position and pose for individual limbs.

The goal of this approach is to provide plausible locations for human limbs. Accurate localization is not required for our application. Likewise, we can tolerate both false positives and false negatives. These bottom up measurements provide a crude proposal distribution that to a more powerful Bayesian inference method for determining 3D human pose and motion [12]. Our experiments have shown that our limb detection algorithm not only enables us to automatically initialize the Loose-limbed spatio-temporal model but also improves the performance of the tracker.

1.1. Previous Work

Recently, the vision community seems to have reached a consensus on the benefits of using cues from several low-level disaggregated models for initialization and robust tracking [8, 12, 13]. These models rely on rapidly segmenting the human shape from the background for detection. Felzensawalb and Huttenlocher have used matching in pictorial structures accompanied by a simple appearance model to recognize human poses in 2D [4]. Wren *et al.* [15] segment the image into blobs using statistical models of the foreground and the background and subsequently use a 2D "Pfinder" model for tracking. Sidenbladh and Black employ robust parameter estimation techniques to differentiate limbs from the background using the image statistics obtained from edge filter, ridge filter and motion cues [10]. In these approaches most noticeable is the work by Cheung *et al.* [1, 2] where they generate shape from silhouettes (SFS). They first generate silhouette images and consequently determine the human form through the visual hulls created by the silhouette images from multiple cameras at the same instant [1, 2]. Similar in concept is the approach in [3], where the authors carry out voxel carving followed by the removal of the pose non-linearities to obtain a skeleton feature for tracking. All the above mentioned models can be used to obtain a proposal distribution of 3D poses. This information from part detectors could be incorporated in various Bayesian Tracking schemes such as Kalman filtering, particle filtering, belief propagation, or Hidden Markov Models (HMM).

In [8, 12, 13] the use of "bottom-up" detectors for automatic initialization of the human model have been reported. Using bottom-up information results in more robust tracking by recovering from occlusion and drift [8]. This project is primarily aiming at devising "bottom-up" 3D limb detectors that exploit ideas from space carving and multi-view eigen models. The resulting estimates of limb location will be used to form a proposal distribution for Bayesian pose estimation [12].

2. Classification

Our goal is to learn simple models of limb appearance which can be used for detection. To that end, we first exploit training data to build a novel multi-view eigen representation. We then exploit this to develop a probabilistic image likelihood model that can be used for detection/localization. Here we learn the eigen model for a single subject. We discuss more general extensions in Section 5.

2.1. Multi-View Eigen Imagery

Our training data consisted of 500 frames from four calibrated cameras where the subject (wearing no special clothing) is walking in a circular trajectory. In addition to the camera images we obtained ground truth estimates of the 3D limb poses from a commercial motion capture system.

For each limb at a given time instant a composite image containing the views from each camera was constructed; that is, for the four $n \times m$ images corresponding to the four cameras a single $4n \times m$ image was built. The composite images are shown in Figure 1(a) for head and left calf. The image regions were independently contrast normalized to lie between 0 and 1 over all the frames. This is done to ensure that Principal Component Analysis captures variations over features and the orientation of the limbs, instead of illumination changes.

Training data are also captured for arbitrary background regions that do not contain human limbs (Figure 1(a),bottom).

2.2. Learning

Once the multi-view training images have been constructed, the images are then rasterized in a lexicographical ordering to build a vector x^t of size $4n \times m$ for every frame t. We subtract the mean image μ from x^t.

$$\mu = \frac{1}{N}\sum_{t=1}^{N} x^t. \tag{1}$$

Then we form a matrix A by concatenating the t mean-subtracted feature vectors $\tilde{x}^t = x^t - \mu$ corresponding to each frame:

$$A = [\tilde{x}^1, ..., \tilde{x}^N]. \tag{2}$$

We perform PCA on the matrix A which is equivalent to a linear transformation $y = T(x) : R^N \rightarrow R^M$. This reduces the feature space to a lower-dimensional subspace, that still accounts for most of the variance. Since the new features obtained after PCA are linear functions of old features, the principal component features can be viewed as the projection of the mean-normalized image vector x^t on the principal eigen vectors ϕ_M, $y = \phi_M^T \tilde{x}$.

Thus by using Principal Component Analysis (PCA) we need significantly fewer components to describe the variance over the features. We take the first M basis vectors that account for 80 percent of the variance. For example, for the head, M = 9 out of the 500 basis vectors. The first four eigen heads and calves are shown in Figure 1 (b, c). As expected, the first few eigen vectors for calves look like filters detecting parallel edges with varying orientations.

2.3. Likelihood Model

The probabilistic likelihood model $P(x|\Omega)$ where x is the $4n \times m$ image vector and Ω is the feature class (body parts in our case) is assumed to be a Gaussian. Therefore the likelihood that a certain image vector is a limb can be represented in the terms of the mean μ and covariance Σ as suggested by Moghaddam and Pentland [7]

$$P(x|\Omega) = \frac{exp[-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)]}{2\pi^{1/2}|\Sigma|^{1/2}}. \tag{3}$$

Let $d(x) = (x-\mu)^T\Sigma^{-1}(x-\mu)$ be the Mahalanobis distance. The term can be rewritten and expressed in terms

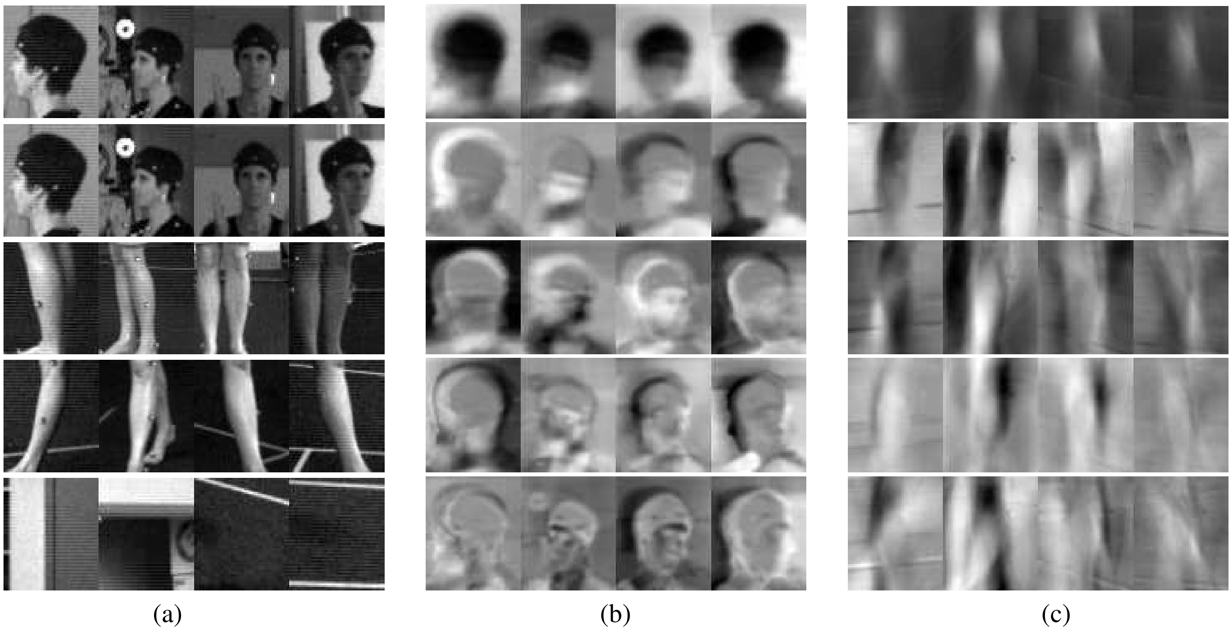


Figure 1. (a) Row 1 and 2 are composite training head images. Similarly Row 3 and 4 are the multi-view calf images. Row 5 is an instance of a multi-view background image. (b) First row is the multi-view mean head image, μ_{head}. The next four rows are $4n \times m$ eigen heads. (c) First row is the multi-view mean left calf image, μ_{calf}. The next four rows are $4n \times m$ eigen calves.

of the projections of the images on the feature subspace (principal eigen vectors). This is done by carrying out eigen decomposition and simple matrix manipulations as described in [7]

$$d(x) = \tilde{x}^T \Sigma^{-1} \tilde{x} = y^T \Lambda^{-1} y. \tag{4}$$

Note that Λ in the diagonal matrix of eigenvalues and hence the Mahalanobis distance can be simply expressed as:

$$d(x) = \sum_{t=1}^{N} \frac{y_t^2}{\lambda_i}. \tag{5}$$

A good approximation of $d(x)$ based on the first M principal components and taking a good statistical estimate of the remaining $N-(M+1)$ components is given by [7]:

$$d(x) = \sum_{t=1}^{M} \frac{y_t^2}{\lambda_i} + \frac{1}{\rho}[\sum_{t=M+1}^{N} y_t^2] \tag{6}$$

where ρ is obtained by minimizing the Kullback-Leibler divergence which yields:

$$\rho = \frac{1}{N-M} \sum_{t=M+1}^{N} \lambda_t \tag{7}$$

The second term in Equation 6 is called the "Distance from Feature Space (DFFS)" [7] which can also be defined as the residual error in approximating the feature space by only M principal components

$$\sum_{t=M+1}^{N} y_t^2 = ||\tilde{x}||^2 - \sum_{t=1}^{M} y_t^2. \tag{8}$$

As can be seen from the above equation we can calculate the DFFS term very efficiently by subtracting the sum of the projections of the test image on the M principal vectors from the L2-Norm $||\tilde{x}||^2$.

2.4. Recognition

A good classifier should distinguish between limbs and general background regions. This implies that not only the classifier should have high likelihood if the image belongs to the object class (limbs) and low likelihood for non-object (background) regions. Consequently we exploit the ratio of the likelihoods of the object and non-object class, where the likelihoods are defined as in Equation 3.

$$LR(x) = \frac{P(x|\Omega = limb)}{P(x|\Omega = background)}. \tag{9}$$

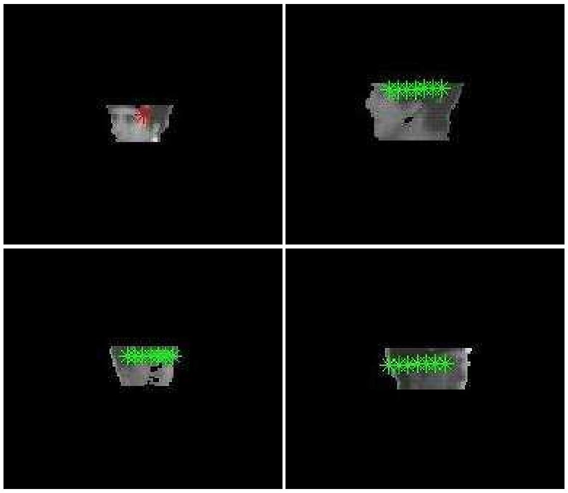


Figure 2. Results of the space carving algorithm. Each of the images are obtained by taking the dot product of the silhouettes with the actual images. Red marker is the randomly chosen point on the first silhouette. Green makers are the projections of the 3D points along the ray (5cm apart).

3. 3D Position and Pose Detection

In this section we describe a search algorithm that helps us obtain a distribution of plausible 3D limb locations. To achieve this goal first we use the motion capture data to learn a limb prior probability distribution. We further confine the search space by exploiting the silhouette images. We uniformly sample from this space to generate a set of plausible 3D poses. Subsequently, we use the likelihood model (as described in previous section) to compute a *maximum a posteriori* (MAP) estimate on the set of plausible poses to obtain the 3D pose of the limb.

3.1. Silhouette Images

The Silhouette images were generated by an adaptive mixture background model. A sequence of background images from the training set were used to learn the mean and the variance in intensity/color of the background pixels over time. The background was then modeled using a mixture of three Gaussians for every pixel. The estimation of the parameters for the Gaussian Mixture model was done by an Expectation Maximization (EM) algorithm. The likelihood of a foreground pixel was assumed to have a uniform distribution over all gray scale values. Finally a simple classifier $P(foreground) \geq P(background)$ was used to identify the foreground pixels.

3.2. 3D Position Detection

We now describe the algorithm for determining the 3D coordinates of the limbs. In order to confine the volume of search space we define a bounding volume for each limb. The ground truth motion data (obtained from a commercial motion capture system) is used to define the bounding volumes. The bounding volumes are defined by the maximum and minimum over all the xyz-coordinates a particular limb traverses in the training movie sequence. This can be interpreted as defining a crude prior probability distribution over the limb locations.

Having defined the bounding volume, we then chose a random point on the foreground of one of the silhouette images for a given time instant. Since we know the extrinsic and intrinsic parameters of the cameras, we know the correspondences between the 3D geometry in the world coordinates and the 2D geometry in the image coordinates. Thus the selected point on the silhouette corresponds to a 3D ray in the world coordinates. The 3D ray is intersected with the bounding volume that defines a right-angled parallelepiped. The right-angled parallelepiped can be viewed as comprising of six rectangles and each of the six rectangles are examined for intersection with the 3D ray sequentially. The problem of solving a Ray-Rectangle Intersection is equivalent to solving two Ray-Triangle Intersection problems. The two triangles are simply obtained by dividing the rectangle along one of the fixed diagonals. The details of the Ray-Triangle Intersection implementation can be found in [9].

Excluding some special cases, we always find two faces of the parallelepiped that intersect with the 3D ray using the above-mentioned methodology. This confines the 3D ray to a line segment. We find the mid point of the line segment and also define the extent to traverse on the line segment. The extent is simply the distance between the two intersection points of the ray and the 3D bounding box. For faster traversal we move fixed steps (distances) from the mid point of the line in both directions and project it on all the remaining silhouettes. If the projection of the point lies on the foreground in all the images the point is a good candidate for the limb location. The algorithm is shown below:

1. Chose a random silhouette image out of the four camera views.
2. Randomly sample a point P on the foreground of the silhouette image.
3. Define 3D Ray in world coordiantes passing through P.
4. Intersect 3D Ray with the bounding volume to get 3D line segment
5. **For** t = -Extent:stepSize:Extent

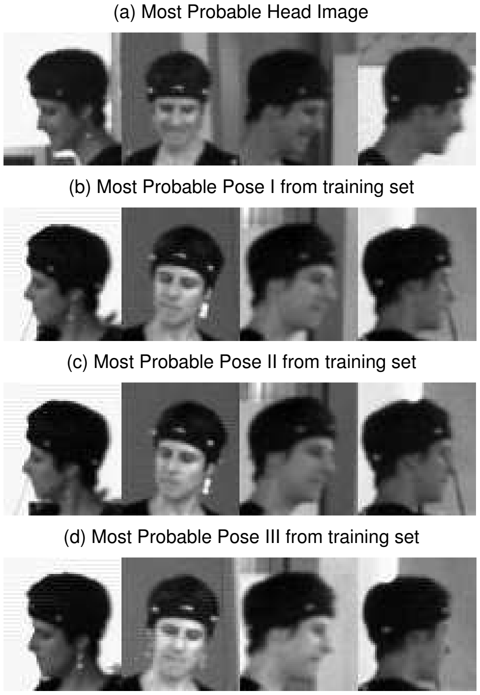
```
        If Project(Ray(t)) is a foreground pixel in all images
            then interestPoint(k) = Ray(t)
                Img = CropImage(interestPoint(k))
                evaluate and store LR(Img)
                k = k+1;

6. goto Step1
```

The aforementioned algorithm is an approximate search method. An arbitrary precision search can be implemented efficiently using well known optimization techniques such as gradient descent, given some relatively weak assumptions on the smoothness of the underlying likelihood surface. The results of the implementation of the algorithm are shown in Figure 2 for head detection. The red marker in the first image is the random point chosen on the silhouette from one of the cameras. The green markers are the projections of the 3D point as we traverse along the 3D line segment. It can be seen form the figure that only points that lie on the silhouettes in all the images are selected as possible locations of the head.

On implementation of the search algorithm we obtain a set of plausible 3D limb locations. We project the point on each of the four camera images and crop a fixed region centered on the projected point in each image. The images from each camera view are then concatenated to generate a multiview image around a plausible 3D limb location. Finally we evaluate the likelihood ratio as described in Section 2.4 for each image. The most probable 3D limb location corresponds to the image that maximizes the likelihood function defined in Equation 9.

3.3. Pose Estimation

The ground truth 3D pose parameters (the orientation of the limb with respect to each axis) were determined for every training frame with the help of the commercial motion capture system. Also the projections (linear coefficients in the eigen space) of the training images on the prinicipal eigen vectors are calculated. We construct a multiview most probable image by cropping fixed regions around the projections of most probable 3D location in each camera view. Subsequently, the Euclidean distance between the linear coefficients of the most probable image and the training images is evaluated. The pose parameters corresponding to the training image whose coefficients have the minimum distance to the coefficients of the most probable image is the estimated limb pose. The results of head pose determination are shown in Figure 3. Similar results were obtained for poses for calves and upper arm detection.


(a) Most Probable Head Image

(b) Most Probable Pose I from training set

(c) Most Probable Pose II from training set

(d) Most Probable Pose III from training set

Figure 3. Results of the Pose Estimation algorithm. (a) Image around the most probable 3D location of head, as predicted by our algorithm. (b), (c), (d) are the images in the training set whose projections in the eigen subspace are least distant to the projections of the image in (a).

4. Results

The results of the “bottom-up” limb detection algorithm at two different time instants are shown in Figure 5 and Figure 6. The colored quadrilaterals correspond to the projection of a tapered cylinder (for each limb) defined by the fixed anatomical parameters of the subject supplemented with the 3D position and pose prediction of our detection algorithm. In Figure 5 and Figure 6 we show the best 25 estimates of the limb positions and poses. Since the eigen model detection is more discriminative across the limbs rather than along the limbs we observe most of the variance in our estimates to be along the axis of the limbs.

Also our limb detection does not distinguish sharply between the left-limbs and the right-limbs; this is shown in Figure 5 and Figure 6. This should be expected because the eigen space for the left and the right side of the body (for the same limbs) is similar and hence our likelihood model does not differentiate between them.

Our multiview eigen space detection algorithm yields accurate results for head and calves, but the results for lower arms are noisy. This can be attributed to occlusion and lack of texture information in case of lower arms. The lack of feature information is reflected in the significant increase in the bases required to account for 80 prercent of the variance for lower arms (M = 23 for lower arms compared to M = 9 for head) in our eigen model. This suggests that an edge based likelihood representation could be a better alternative for detecting lower arms [7].

Figure 4 shows accuracy of automatic initialization for the 3D body model, based on the limb proposal distributions formed using the limb detection results. We employ Loose-limbed spatio-temporal model [11] defined over two consecutive frames for this purpose. Pose estimation and tracking is defined as inference over this 3D spatio-temporal body model and is carried out using non-parametric Belief Propagation algorithm. For further details on the body-model, its automatic initialization using limb shouters, and the inference algorithm we refer readers to [11].

In [11] a metric for quantitative evaluation of accuracy based on the absolute distance to true and estimated markers locations on the limbs was presented. We use this distance-based error measure here to evaluate the convergence properties of the automatic initialization based on the full, and sub-sets of proposal distributions formed using the results of detection algorithm. Limbs not initialized using the proposal distributions are initialized by uniformly sampling the space of all possible position and orientations in 3D space. Results reported in Figure 4 are averaged over 5 independent runs (due to the stochastic nature of the algorithm), and over the two consecutive frames for which the spatio-temporal model was defined.

One can see that the model converges faster and to a more accurate solution when more proposals for different limbs are available. It can also be seen that the largest performance increase can be attributed to the head detection. Head detection tends to be most accurate, and unlike other limb detectors considerably more robust in its orientation estimates (due to the rich and non-uniform texture). Over all the experiment in Figure 4 shows that the proposed detection algorithm is sufficient for initializing and tracking a person in 3D using the framework proposed in [11, 12].

5. Summary and Conclusions

In this paper, we have introduced a method for "bottom-up" 3D limb detectors. The limbs are detected using an approach that merges the ideas of space carving and eigen recognition. We have applied the algorithm in detecting head, upper arms, lower arms and calves in over 300 frames. Our multi-view eigen approach deals with occlusion to a certain extent, but modeling occlusion explicitly through depth-maps and ray-tracing should improve the results significantly. The current methodology, as is can be used to generate 'shouters' for other higher level tracking systems based on belief propagation [12]. The results can be improved by employing classifiers that are scale, rotation and illumination invariant. Currently we are working to implement more advanced learning algorithms like AdaBoost for limb detection [14]. This approach combines several weak classifiers in a cascaded-fashion which carry out the background/foreground classification and very efficiently narrow down the search to the object of interest. Also in the future we plan to train our PCA model on a richer training set by including images of multiple subjects. This would help us to extend our approach for detecting multiple persons in a scene.

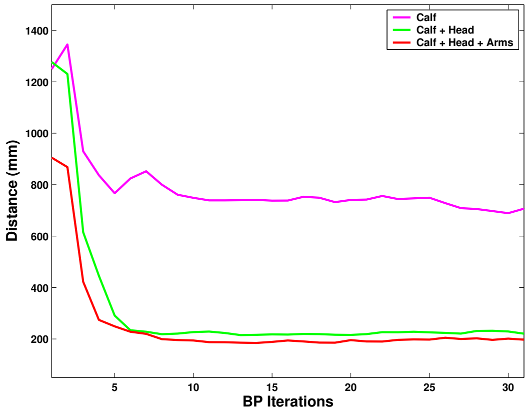


Figure 4. Quantitative evaluation of initialization error as a function of iterations of the belief propagation algorithm. Limbs that are not initialized from the proposal, are initialized using a set of samples drawn form a uniform distribution defined over limb positions and orientations in 3D.

Acknowledgments

This work was supported by the DARPA HumanID program ONR contract N000140110886. We thank Stefan Roth for his insightful suggestions during the course of the project.

References

[1] G. Cheung, S. Baker, and T. Kanade. Shape-from-silhouette of articulated objects and its use for human body kinematics estimation and motion capture. *ICCV*, 1:77–84, 2003.

[2] G. Cheung, T. Kanade, J. Bouguet, and M. Holler. A real time system for robust 3D voxel reconstruction of human motions. *ICCV*, 2:714–720, 2000.

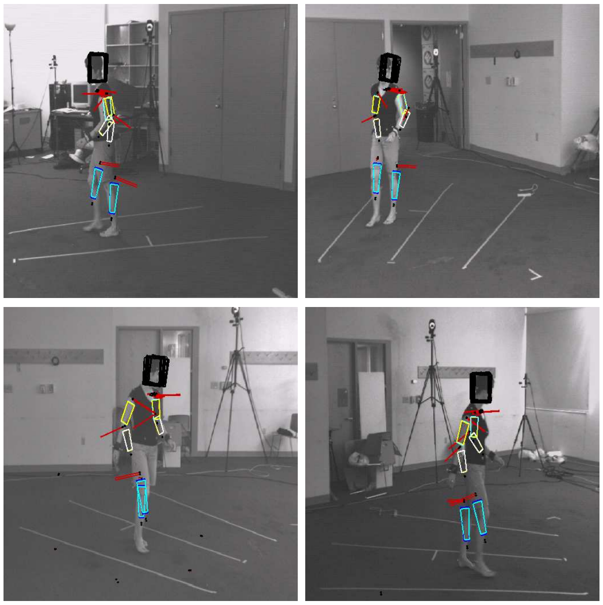


Figure 5. Top 25 Detection estimates of our limb detector algorithm. Each image corresponds to a different camera view of the same scene. The limbs are color coded Head: Black, Left Calf: Cyan, Right Calf: Blue, Left Upper Arm: Green, Right Upper Arm: Yellow, Left Lower Arm : White, Right Lower Arm : Yellow

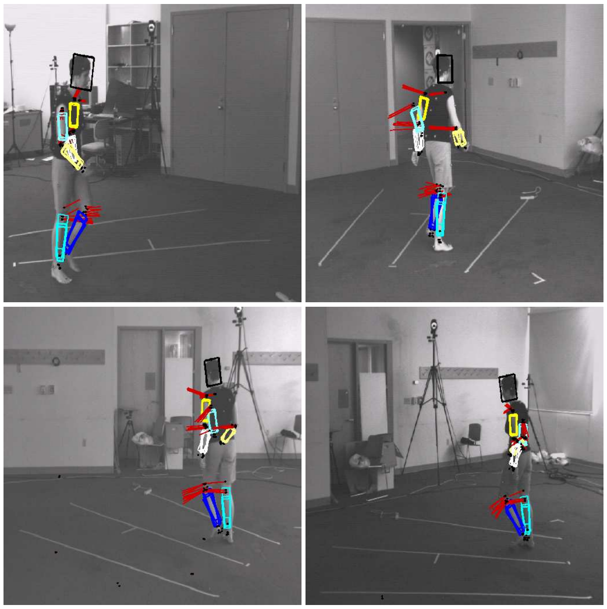


Figure 6. Top 25 limb position and pose estimates for frame 305. The limbs are color coded Head:Black, Left Calf:Cyan, Right Calf: Blue, Left Upper Arm: Green, Right Upper Arm: Yellow, Left Lower Arm : White, Right Lower Arm : Yellow. The red lines represent the x-axis in the limb co-ordinate system.

[3] C. Chu, O. Jenkins, and M. Matarić. Markerless kinematic model and motion capture from volume sequences. *ICCV*, II:475–482, 2003.

[4] P. Felzenszwalb and D. Huttenlocher. Efficient matching of pictorial structures. *CVPR*, 2:66–73, 2000.

[5] S. Ioffe and D. Forsyth. Human tracking with mixture of trees. *ICCV*, I:690–695, 2001.

[6] M. Isard and A. Blake. Condensation – conditional density propagation for visual tracking. *IJCV*, 29:2–28, 1998.

[7] B. Moghaddam and A. Pentland. Probabilistic visual learning for object detection. *Proceedings in 5th International Conference on Computer Vision*, pages 786–793, 1995.

[8] D. Ramanan and D. Forsyth. Finding and tracking people from bottom up. *CVPR*, II:467–474, 2003.

[9] P. Schneider and D. Eberly. *Geometric Tools for Computer Graphics*. Morgan and Kaufmann, 2003.

[10] H. Sidenbladh, M. Black, and L. Sigal. Learning image statistics for Bayesian tracking. *ICCV*, 2:709–716, 2001.

[11] L. Sigal, S. Bhatia, S. Roth, M. Isard, , and M. J. Black. Tracking loose-limbed people. *CVPR*, 2004.

[12] L. Sigal, M. Isard, B. Sigelman, M. J. Black, and D. A. Forsyth. Attractive people: Assembling loose-limbed models using non-parametric belief propagation. *NIPS*, 2003.

[13] L. Taycher, J. W. F. III, and T. Darrell. Combining simple models to approximate complex dynamics. *SMVP*, 2004.

[14] P. Viola and M. Jones. Robust real-time face detection. *ICCV*, II:747, 2001.

[15] C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland. Pfinder: Real time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:780–785, 1997.